

Initiation au logiciel SAS

M1 MAS - Année universitaire 2023-2024

Alexie Robert

29 septembre 2023

Sommaire

- 1 Points clés de la séances du 29 septembre et retour sur le TP
- 2 Ce que l'on va aborder aujourd'hui

Les objectifs du TP précédent

- Représenter des graphiques avec les procédures GCHART, GPLOT et BOXPLOT ;
- enregistrer les sorties avec l'Output Delivery System ;
- exporter vers des fichiers Excel ;
- utiliser les procédures MEANS, UNIVARIATE, CORR et FREQ ;
- découvrir les instructions ARRAY et les boucles ;

Les objectifs du TP du 6 octobre

- Découvrir les macro-variables : %LET et CALL SYMPUT ;
- comprendre la logique des programmes macro ;
- découvrir les instructions %EVAL et %INCLUDE ;
- découvrir les fonctions UPCASE, TRANWRD, INDEX, RANUNI, SUBSTR, CATX ;
- découvrir les PROC SUMMARY et TRANSPOSE.

Les macro-variables

1 le programme

```
OPTIONS SYMBOLGEN;  
/* Création de la macrovariable somme à laquelle on affecte  
la valeur Je suis comique */  
%LET somme=Je suis comique;  
  
/* Dans le journal s'affiche la valeur de la macrovariable somme */  
%PUT &somme;
```

Les macro-variables

1 le programme

```

OPTIONS SYMBOLGEN;
/* Création de la macrovariable somme à laquelle on affecte
la valeur Je suis comique */
%LET somme=Je suis comique;

/* Dans le journal s'affiche la valeur de la macrovariable somme */
%PUT &somme;

```

2 dans le Journal

```

1      %LET somme=Je suis comique;
2
3      /* Dans le journal s'affiche la valeur de la macrovariable somme */
4      %PUT &somme;
Je suis comique
5      OPTIONS SYMBOLGEN
6      %LET somme=Je suis comique;
_

```

Les macro-variables

1 le programme

```
%LET somme=5783.4;
```

```
▢ DATA herp;  
  set TP.herp;  
  PART=POIDS/&somme;  
RUN;
```

Les macro-variables

1 le programme

```
%LET somme=5783.4;
```

```

DATA herp;
  set TP.herp;
  PART=POIDS/&somme;
RUN;
```

2 dans le Journal

```

20  %LET somme=5783.4;
21
22  DATA herp;
23  set TP.herp;
24  PART=POIDS/&somme;
SYMBOLGEN:  La macro-variable SOMME est résolue en 5783.4
25  RUN;
```


Les macro-variables

1 le programme

```
%LET somme=5783.4;
```

```
DATA herp;
```

```
set TP.herp;
```

```
PART=POIDS/&somme;
```

```
RUN;
```

2 dans le Journal

```
20 %LET somme=5783.4;
21
22 DATA herp;
23 set TP.herp;
24 PART=POIDS/&somme;
SYMBOLGEN: La macro-variable SOMME est résolue en 5783.4
25 RUN;
```

2 dans la table (extract)

VIEWTABLE: Work.Herp			
	IDEN	POIDS	PART
1	I0004	79.7	0.013780821
2	I0008	91.4	0.0158038524
3	I0009	79.5	0.0137462392
4	I0011	70.9	0.0122592247
5	I0012	84.6	0.0146280735
6	I0013	86.7	0.0149911817
7	I0015	95	0.0164263236

L'instruction `_NULL_`

On peut l'utiliser pour affecter une valeur à une macro-variable.

L'instruction `_NULL_`

On peut l'utiliser pour affecter une valeur à une macro-variable.

1 le programme

```

DATA _NULL_;
  SET tp.herp END=fin;
  RETAIN SOMME 0;
  SOMME=SOMME+POIDS;
  if fin=1 THEN CALL SYMPUT("sommenew", SOMME);
RUN;
%PUT &sommenew;

```

L'instruction `_NULL_`

On peut l'utiliser pour affecter une valeur à une macro-variable.

1 le programme

```

DATA _NULL_;
  SET tp.herp END=fin;
  RETAIN SOMME 0;
  SOMME=SOMME+POIDS;
  if fin=1 THEN CALL SYMPUT("sommenew",SOMME);
RUN;
%PUT &sommenew;

```

2 dans le Journal

```

SYMBOLGEN:  La macro-variable SOMMENEW est résolue en          5783.4
33  %PUT &sommenew;
5783.4

```

Les macro-programmes

- 1 un programme pour réaliser un graphique

```
❏ PROC GPLOT DATA=herp;  
  PLOT sys*dia;  
RUN;
```

Les macro-programmes

1 un programme pour réaliser un graphique

```
❏ PROC GPLOT DATA=herp;  
   PLOT sys*dia;  
RUN;
```

2 sa transformation en macro-programme

```
/* Création d'un macro-programme pour dessiner des graphiques */  
%MACRO plot(tab=, yvar=, xvar=);  
PROC GPLOT DATA=&tab;  
PLOT &yvar*&xvar;  
RUN;  
%MEND plot;
```

Les macro-programmes

- 1 un programme pour réaliser un graphique

```
❏ PROC GPLOT DATA=herp;
  PLOT sys*dia;
  RUN;
```

- 2 sa transformation en macro-programme

```
/* Création d'un macro-programme pour dessiner des graphiques */
%MACRO plot(tab=,yvar=,xvar=);
PROC GPLOT DATA=&tab;
PLOT &yvar*&xvar;
RUN;
%MEND plot;
```

- 2 et l'exécution avec les paramètres

```
%plot(tab=tp.herp,xvar=DIA,yvar=SYS);
```

La fonction UPCASE

1 le programme

```
/* Création de la variable id2 qui met en majuscule la variable id */  
data tel2;  
set tel;  
id2=upcase(id);run;  
proc print data=tel2;run;
```


La fonction UPCASE

1 le programme

```
/* Création de la variable id2 qui met en majuscule la variable id */
data tel2;
  set tel;
  id2=upcase(id);run;
proc print data=tel2;run;
```

2 la sortie

Obs.	id	tel	ordre	id2
1	albert	0467780614	3	ALBERT
2	albert	0610227491	2	ALBERT
3	albert	0452364187	1	ALBERT
4	beatrice	0552201456	1	BEATRICE
5	beatrice	0145254631	2	BEATRICE
6	cedric	0235572076	1	CEDRIC
7	daniele	0325897456	1	DANIELE

La procédure TRANSPOSE

1 le programme

```
/* Transposition de la table tel. On obtient une ligne par id et  
une colonne par numéro de téléphone */
```

```
proc sort data=tel; by id ordre; run;
```

```
proc transpose data=tel out=b prefix=ntel;  
by id;  
var tel;  
run;
```

La procédure TRANSPOSE

1 le programme

```
/* Transposition de la table tel. On obtient une ligne par id et
une colonne par numéro de téléphone */
```

```
proc sort data=tel; by id ordre; run;
```

```
proc transpose data=tel out=b prefix=ntel;
  by id;
  var tel;
run;
```

2 la sortie

Obs.	id	_NAME_	ntel1	ntel2	ntel3
1	albert	tel	0452364187	0610227491	0467780614
2	beatrice	tel	0552201456	0145254631	
3	cedric	tel	0235572076		
4	daniele	tel	0325897456		